Autonomic Computing via Dynamic Self-Repair of Hardware Faults

First Annual Report: January 1, 2005 - January 1, 2006

PI Daniel J. Sorin, co-I Sule Ozev

Department of Electrical and Computer Engineering

Duke University

1 Introduction

The goal of this research project is to develop computer systems that use dynamic self-repair to enable *autonomic operation in the presence of permanent hardware faults*. Autonomic operation is crucial for reliability when mission-critical computer systems are deployed for a long time with little or no opportunity for human repair. Autonomic operation is also becoming more important for commercial computer systems, as both the number of computers in an installation and the hardware fault rates continue to rise. Autonomic systems detect errors, diagnose their underlying faults, and take action to mask hard faults. In this research proposal, we focus on microprocessors, since they comprise the majority of computational infrastructure, although we plan to apply our techniques more broadly (e.g., to network processors, digital controllers, etc.).

As microprocessor fabrication technology continues to shrink devices and wires and increase clock frequencies, hard fault rates in hardware are consequently increasing. For example, with decreasing feature sizes, the mean time to breakdown for interconnects due to electromigration decreases [7], making hard faults more likely. As another example, shrinking transistor gate oxide thickness also decreases the lifetime of transistors under continuous operational stress [8, 6]. These effects are exacerbated by stressful operating conditions, such as those experienced by computers that operate outside of the protection of Earth's atmosphere. Moreover, with increasing numbers of transistors being used, the probabilities of microprocessor hard faults are correspondingly increasing.

Existing solutions for tolerating hard faults are either very expensive—in terms of power and hardware cost—or suffer unacceptable performance penalties for many classes of hard faults. One class of approaches uses redundant parallel processors (e.g. triple modular redundancy) to provide forward error recovery (FER). These systems provide high reliability and performance, but they use large amounts of power and hardware. At the other end of the spectrum, a recently developed backward error recovery (BER) scheme, called DIVA [2], uses only a small on-chip checker to achieve almost as much reliability as redundant processor schemes. DIVA uses much less hardware and power, but it incurs significant perfor-

mance and energy penalties for recovery every time a fault is exercised. These penalties can be particularly problematic for hard faults in heavily used circuits, such as the reorder buffer.

In this research project, we seek to develop lightweight hardware techniques for self-repair that can achieve high reliability and robust performance without consuming vast amounts of power or hardware.

2 Achievements to Date and Plans for Continuing Research

In this section, we discuss what we have achieved thus far and what we plan to continue working on. We present this in the framework of the research plan that was in our original proposal (original plan in *italics*).

- *We will improve our initial implementation of Self-Repairing Array Structures (SRAS) and evaluate its potential to protect a variety of array structures within modern microprocessors.*

We have largely achieved this goal. First, we extended SRAS [4] by creating a second implementation of it that uses error detecting codes (EDC) instead of a more complicated approach for detecting errors. We have submitted a paper on this work to IEEE Transactions on Dependable and Secure Computing.

Second, we developed an integrated approach to detecting, diagnosing, and reconfiguring around hard faults in microprocessor structures. This research has been accepted for publication at the highly selective (10-20% acceptance rate) 2005 International Symposium on Microarchitecture [5], and it will be presented at the conference by a graduate student in our research group. The key to our scheme is the diagnosis mechanism. As each instruction traverses the microprocessor pipeline, it records which structures it has used (e.g., ALU#2, reorder buffer entry #23, and load/store queue entry #17). If the instruction is detected to be faulty, the error counters for each of these components is incremented. If any of the counters exceeds its threshold, it indicates a hard fault, since transient faults will never lead to above-threshold counters (we clear the counters periodically). In conjunction with this diagnosis mechanism, we use an existing technique for error detection/correction, DIVA [2], and new and existing techniques for reconfiguring the processor to avoid using faulty components.

- *We will develop detailed hard fault models that correspond to the underlying physical phenomena that occur in modern VLSI technology.*

We developed a high-level hard fault model for a 64-bit adder in a microprocessor for use in our MICRO paper, and we are currently working to extend this to other structures. The challenge is developing a model that is both detailed enough but still tractable in simulation. On a related topic, we are developing a new metric for evaluating a processor's resilience to hard faults. This is an extension of prior work that developed the Architectural Vulnerability Factor (AVF) that applies only to transient faults [3]. The existing AVF metric provides misleading results if naively applied to hard fault scenarios.

- *We will fully develop and evaluate Hierarchical Modular Redundancy (HMR), including efficient fault detection and remapping schemes.*

We have begun this research, by exploring HMR for n-bit adders and n-bit multipliers. The initial results are promising. These fault-tolerant adders and multipliers can tolerate more hard faults than existing schemes (including full replication) with less hardware cost and less power consumption. We are in the process of integrating these designs into the microprocessor simulator to explore their impact on the overall behavior. We will soon perform detailed VLSI layouts of these designs to ensure that they can be fabricated without undue difficulty.

- *We will apply SRAS and HMR to system models other than just microprocessors, including network processors and embedded controllers.*

This research is still in the future.

- *We will, in conjunction with all of the above plans, develop simulation infrastructure for evaluating our ideas—in terms of power, hardware, and performance—at the architectural, circuit, and device levels.*

We have already developed simulation tools for exploring the impact of hard faults on microprocessors. This aspect of the development has been built on top of the industry-standard SimpleScalar simulator [1]. We are working to develop an automated and parameterized method of injecting hard faults into SimpleScalar. The tool development at lower levels (circuits and devices) has thus far mostly consisted of using software scripts to incorporate existing tools such as SPICE.

3 Education and Training

Beyond the obvious research contributions of this project, it has also provided support and infrastructure for the education and training of two graduate students (Mahmut Yilmaz and Bogdan Romanescu) and one undergraduate research assistant (Derek Hower). Derek Hower participates in the research program as part of Duke's prestigious Pratt Research Fellow program.

References

[1] T. Austin, E. Larson, and D. Ernst. SimpleScalar: An Infrastructure for Computer System Modeling. *IEEE Computer*, 35(2):59–67, Feb. 2002.

[2] T. M. Austin. DIVA: A Reliable Substrate for Deep Submicron Microarchitecture Design. In *Proceedings of the 32nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 196–207, Nov. 1999.

[3] A. Biswas, P. Racunas, R. Cheveresan, J. Emer, S. S. Mukherjee, and R. Rangan. Computing Architectural Vulnerability Factors for Address-Based Structures. In *Proceedings of the 32nd Annual International Symposium on Computer Architecture*, June 2005.

[4] F. A. Bower, P. G. Shealy, S. Ozev, and D. J. Sorin. Tolerating Hard Faults in Microprocessor Array Structures. In *Proceedings of the International Conference on Dependable Systems and Networks*,

pages 51–60, June 2004.

[5] F. A. Bower, D. J. Sorin, and S. Ozev. A Mechanism for Online Diagnosis of Hard Faults in Microprocessors. In *Proceedings of the 38th Annual IEEE/ACM International Symposium on Microarchitecture*, Nov. 2005.

[6] K. C. Boyko and D. L. Gerlach. Time Dependent Dielectric Breakdown at 210 Å Oxides. In *Proceedings of the 27th Annual Reliability Physics Symposium*, pages 1–8, Apr. 1989.

[7] Z. Li, G. Wu, Y. Wang, Z. Li, and Y. Sun. Numerical Calculation of Electromigration Under Pulse Current with Joule Heating. *IEEE Transactions on Electron Devices*, 46(1):70–77, Jan. 1999.

[8] B. P. Linder, J. H. Stathis, D. J. Frank, S. Lombardo, and A. Vayshenker. Growth and Scaling of Oxide Conduction After Breakdown. In *41st Annual IEEE International Reliability Physics Symposium Proceedings*, pages 402–405, Mar. 2003.